ZER0GEN: Efficient Zero-shot Learning via Dataset Generation

Anonymous ACL submission

Abstract

There is a growing interest in dataset generation recently due to the superior generative capacity of large pre-trained language models (PLMs). In this paper, we study a flexible and efficient zero-shot learning method, ZER0GEN. Given a zero-shot task, we first generate a dataset from scratch using PLMs in an unsupervised manner. Then, we train a tiny task model (e.g., LSTM) under the supervision of the synthesized dataset. This approach allows highly efficient inference as the final task model only has orders of magnitude fewer parameters comparing to PLMs (e.g., GPT2-XL). Apart from being annotation-free and efficient, we argue that ZER0GEN can also provide useful insights from the perspective of data-free model-agnostic knowledge distillation, and unreferenced text generation evaluation. Experiments and analysis on different NLP tasks, namely, text classification, question answering, and natural language inference, show the effectiveness of ZER0GEN.

1 Introduction

While generating training data with language model is not new to natural language processing (Anaby-Tavor et al., 2020; Puri et al., 2020; Kumar et al., 2020), it has garnered enormous interests recently due to the superior generative capacity of large-scale pre-trained language models (PLMs). Training examples created in such a manner have been found effective in various scenarios via the data augmentation procedure (Lee et al., 2021; Schick and Schütze, 2021; Wang et al., 2021; Meng et al., 2022, inter alia).

In this paper, we study an extreme scenario of such an approach, ZER0GEN. Given a downstream task, we first generate its training data from scratch using a powerful PLM, whose generation is steered by carefully designed task-specific prompts. Then, we train a tiny task model (TAM), which has orders of magnitude fewer parameters than PLMs, under the supervision of the synthesized training data. Machine generated text is the only medium that connects the PLMs to the final task models, and no human annotations are required in the entire process. The TAM can be of any choice (e.g., loglinear or neural), allowing efficient inference\(^1\) and deployment. Besides, TAM can be flexibly designed with any task-specific strategies (e.g., inductive bias or loss), which could provide superior performance.

Apart from being annotation-free and efficient, we are also interested in ZER0GEN for the following reasons. First, ZER0GEN can be seen as a variant of knowledge distillation (KD; Hinton et al. (2015)) that provides some exciting new features. Unlike conventional KD, ZER0GEN does not require any human annotations during distillation. Furthermore, ZER0GEN makes no presumption on the architecture choice of student models, thus we can incorporate any task-specific inductive bias into the design of student models conveniently. Second, ZER0GEN can serve as an unreferenced evaluation method for text generation (Guan and Huang, 2020; Pillutla et al., 2021): the downstream tasks’ performance is dominated by the quality of the synthesized text, thus can serve as an indirect measure of the generation models and algorithms. Third, ZER0GEN sheds new lights on prompt engineering (Petroni et al., 2019; Brown et al., 2020) (i.e., the design of the prompts in PLMs). As manual prompts reflect our essential knowledge of specific tasks, an intriguing question here is to what extent we can incorporate human knowledge or instructions in these prompts.

We evaluate ZER0GEN in three NLP tasks which are text classification, question answering, and natural language inference, across six datasets. Our key research findings are summarized as follows:

\(^1\)Amazon estimates that 90% of production ML infrastructure costs are for inference, rather than training (Jain et al., 2019).
• The zero-shot performance of TAM in ZeroGen framework significantly surpasses its PLM counterparts (which often serves as the teacher models under the knowledge distillation context), with only \( \sim 0.4\% \) number of parameters (§4.2);
• In some low-resourced settings, TAM trained with synthesized data even outperforms the same model trained with human annotations in a fully supervised manner (§4.3);
• The quality of the generated text by known models and algorithms are well reflected in downstream tasks’ performance, and decoding strategies that encourage more diversity also result in greater noise (§4.4);
• Prompt engineering is challenging – the performance of more instructive or natural language style prompts varies in different tasks (§4.5).

In conclusion, we argue that ZeroGen is a viable and promising approach towards flexible and efficient zero-shot learning in NLP. It also has a great potential as a data-free model-agnostic knowledge distillation and unreferenced text evaluation method. Our code can be found at Github\(^2\).

2 Preliminary: Prompt-based Zero-Shot Learning

We start with preliminary knowledge about prompt-based zero-shot learning framework (named PROMPTING). Promoting first instantiates a prompt \( T(\cdot) \) with each input \( x_i \in X \) and outputs a natural language sequence to be completed by \( P \). For instance, we show an example on sentiment analysis task in Figure 1(a), where \( x_i \) is "A deep and meaningful film." and \( T(x_i) \) is "A deep and meaningful film. The sentiment of the movie review is ". Furthermore, Promoting defines a verbalizer \( M(\cdot) \) that maps each label/class \( y_i \) to a word/words in \( P \)’s vocabulary. For instance, "positive" and "negative" represents the two classes. In this way, Promoting models the probability of class \( y_i \in Y \) for \( x_i \) as:

\[
p(p(y_i|x_i) = P(M(y_i)|T(x_i))). \tag{1}
\]

During the whole process, the pre-trained weights of \( P \) are frozen and no training is required.

The vast linguistic (Jawahar et al., 2019; Goldberg, 2019; Tenney et al., 2019) and factual (Petroni et al., 2019; Jiang et al., 2020b) knowledge encoded in PLMs’ parameters is the key towards Promoting’s success. However, Promoting fails to fully exert the capacity of PLMs and heavily relies on gigantic PLMs during inference. This motivate us to explore a more flexible and efficient way of conducting zero-shot learning with PLMs.

3 ZeroGen

In this work, we take the dataset generation method to the extreme and study ZeroGen, a flexible and efficient zero-shot learning framework via dataset generation. ZeroGen framework comprises three sequential stages as shown in Figure 1(b):

1. The goal of the first stage is to make use of the generative power of PLMs to synthesize a dataset to solve the downstream task. With carefully designed prompts and a powerful PLM, the generated dataset is believed to incorporate rich task-specific knowledge.
2. Giving pseudo dataset synthesized as above, we then train a tiny task model (TAM) to solve the task. TAM can integrate with any task-specific inductive bias and is also order-of-magnitude smaller than PLMs.
3. Finally, we perform efficient inference on target task using the trained model. During the whole process, no human annotations are involved, thus the evaluation setting is purely zero-shot.

Pseudo dataset generation For a single-sentence classification task \( D \), we aim to generate a pseudo dataset \( D^g = (X^g, Y^g) \) with the help of a left-to-right PLM \( P \). We first sample a class label \( y^g \) from a uniform distribution:

\[
y^g \sim U(y_1, y_2, \ldots, y_k), \tag{2}
\]

where \( k \) is the number of classes. \( y^g \) is then wrapped up into a label-descriptive prompt \( T(y^g) \) to steer the generation of \( x^g \):

\[
x^g \sim P(x^g|T(y^g)). \tag{3}
\]

Since the parameters of \( P \) is frozen and the generation \( x^g \) for each \( y^g \) is deterministic, we can adopt different sampling algorithms (e.g., Top-k sampling (Fan et al., 2018) and nucleus sampling

\[\text{URL of codes is omitted here pending the review process.}\]
(Holtzman et al., 2020)) to increase the diversity of generated dataset. We then pair the generated $x^g$ with $y^g$ to construct a pseudo training data. We show an example about generating a pseudo sentiment classification dataset in Figure 1(b). The prompt $T(y^g)$ for a positive label $y^g$ is "The movie review in positive sentiment is: "". With the sampling strategies, this prompt steers PLMs to generate multiple sentence ending with another quotation mark, e.g., "A deep and meaningful movie."" or "Good film!!!"".

For sentence-pair classification tasks, we need to generate two sequences that bear certain relationships (e.g., premise and hypothesis in NLI, context and question in QA). We decompose the generation into two steps: (i) We first generate and/or sample a conditional contextual $c^g$ (e.g., $c^g$ represents premise in NLI and context in QA). The context $c^g$ is then concatenated with a sampled label $y_g$ and transformed into a prompt $T(c^g, y_g)$. (ii) Giving the prompt $T(c^g, y_g)$, we can now generate the other sentence $x^g$ (e.g., hypothesis in NLI and question in QA) as in Equation (3). In current implementation, we sample $c^g$ from an unlabeled corpus. But $c^g$ can also be generated following procedure of generation for single-sentence classification task. Since there could be no predefined label set for extractive QA task, we use publicly available spaCy\textsuperscript{3} toolkit to annotate entities, and then uniformly select an entity as $y^g$. Finally, the generated sentence-pair and label can form the pseudo dataset $D^g = (C^g, X^g, Y^g)$. We elaborate details on prompts chosen for each task in Section 4.5.

### Pseudo-supervised training

With the pseudo dataset $D^g$, we train a tiny task model TAM to conduct the given task. This procedure is highly flexible, meaning that we can use any model architecture, loss function, and training strategy. In this work, we primarily focus on the overall framework, thus we leave the tuning of these components for future work. Under the zero-shot learning setting, it should be noted that it is not possible to access the standard validation set. Therefore, we use a portion (e.g., 10%) of the pseudo dataset as the validation set to perform model selection.

### Zero-shot evaluation

Finally, we conduct inference on the trained TAM model. As TAM is order-of-magnitude smaller than PLM, it is able to perform extremely efficient inference.

\textsuperscript{3}https://spacy.io/
Table 1: Evaluation results for ZEROGEN framework at three different scales of PLM and two different scales of TAM. The ZEROGEN results that outperform PROMPTING using the same PLM are in grey, and the best result for each task using the same PLM is bolded. * indicates that the result of TAM under ZEROGEN framework outperforms the same TAM under SUPERVISED framework. Due to the architecture difference in LSTM-based models among different tasks, we calculate the average number of parameters (i.e., 7M).

4 Experiments

4.1 Setup

We perform experiments across three different tasks including six different NLP datasets. The detailed experimental setup (i.e., Evaluation Strategy and Implementation Details) are described in Appendix A.

Datasets  We consider two Text Classification datasets (i.e., SST-2 (Socher et al., 2013) and IMDb (Maas et al., 2011)), two Natural Language Inference datasets (i.e., QNLI (Rajpurkar et al., 2016) and RTE (Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009)), and two Question Answering datasets (i.e., SQuAD1.1 (Rajpurkar et al., 2016) and AdversarialQA (Bartolo et al., 2020)). The number of training examples for SST-2 and RTE is 6.9k and 2.5k, which can be considered as low resource of training examples for SST-2 and RTE is 6.9k and 2.5k, which can be considered as low resource.

Baselines  We compare ZEROGEN framework with two baselines: (1) PROMPTING. The prompt-based zero-shot learning framework via PLMs. We use GPT2 (117M), GPT2-large (762M), and GPT2-XL (1.5B) (Radford et al., 2019) via the HuggingFace Transformers library (Wolf et al., 2019). (2) SUPERVISED. The TAMs are trained on standard dataset (i.e., human annotations). Regarding model architecture of TAMs, we use two types of model for each task: a LSTM-based model (i.e., BiLSTM (Hochreiter and Schmidhuber, 1997) for TC and NLI tasks, and BiDAF (Seo et al., 2017) for QA task), and a tiny pre-trained model (i.e., DistilBERT (Sanh et al., 2019)).

4.2 ZEROGEN vs. PROMPTING

Table 1 compares ZEROGEN with PROMPTING framework. We observe that ZEROGEN significantly outperforms PROMPTING on most datasets we evaluated, and this superiority is consistent across different PLM generators and TAMs. In particular, when using DistilBERT as TAM, we find that among 18 (3 generators × 6 tasks) head-to-head comparison with PROMPTING, ZEROGEN achieves better performance in 15 cases4. The reasons for the superior performance are mainly two-folds: 1) compared with the general purpose generation model, task-specific classification model may encourage a more deterministic decision boundary, which shares the same spirits with entropy minimization (Grandalet and Bengio, 2006) or self-training (Lee et al., 2013), and 2) classification tasks benefit from the inductive bias in the architecture. For example, method that predicts the start and end positions greatly narrows down the searching space for extractive question answering tasks, in comparison with free-generation on the vocabulary space.

Besides the superior effectiveness in zero-shot learning, it’s also worth noting that ZEROGEN is also quite efficient. ZEROGEN can achieves comparable (LSTM) and even better (DistilBERT)
performance than PROMPTING, using more than 200 times and 20 times fewer parameters, respectively. Nowadays, with increasingly larger pre-trained language models (e.g., 175B GPT-3 (Brown et al., 2020), 1571B Switch-C (Fedus et al., 2021)), the advantage of ZEROGEN becomes even more pronounced. The gigantic PLMs can improve the quality of synthesized dataset and lead to better zero-shot performance. Meanwhile, the TAM can remain light-weighted for efficient inference and serving.

Furthermore, when scaling up PLMs, we observe continuous performance boost for both PROMPTING and ZEROGEN. This indicates that large-scale PLMs might have been trained to store more knowledge that is useful for generating accurate dataset for a task.

### 4.3 ZEROGEN vs. SUPERVISED

It’s commonly accepted that the zero-shot performance of a NLP model can lag way behind its fully-supervised performance (trained on human annotations). However, we find that ZEROGEN even outperforms its SUPERVISED counterpart on SST-2 and RTE datasets (highlighted with * in the Table 1). Our conjecture is the size of the datasets are the key factor. ZEROGEN automatically generates much more data as supervision during training (i.e., 200k synthesized samples vs. 6.9k/2.5k human annotations). These results are encouraging because they suggest that: (i) ZEROGEN is quite effective in low-resource scenario; (ii) it’s possible to synthesize training samples to approximate human-annotations in a fully unsupervised manner.

We further investigate if we can trade data volume in exchange for zero-shot performance in ZEROGEN. Our results are shown in Figure 2.

Overall, we find that the final performance improves continuously as the amount of data grows, despite diminishing returns. We find that generating 10k of training samples leads to better performance than PROMPTING method on most datasets. In addition, by increasing data size, we find that ZEROGEN even outperforms the SUPERVISED baseline on SST-2 and RTE. But still, on some datasets examined (e.g., SQuAD, QNLI), there remain a performance gap between ZEROGEN and SUPERVISED.

### 4.4 ZEROGEN as Text Generation Evaluator

The quality of the synthesized text is the key to the performance of the downstream tasks. ZEROGEN can thus be seen as an indirect measure of the generation models and algorithms. It is
Table 2: Results for comparing different decoding methods with selected parameters of each method. The best result under different decoding methods is bolded.

<table>
<thead>
<tr>
<th>TAM</th>
<th>Strategy</th>
<th>IMDb</th>
<th>SQuAD</th>
<th>QNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT-2</td>
<td>Greedy</td>
<td>74.40</td>
<td>24.19/31.16</td>
<td>63.59</td>
</tr>
<tr>
<td></td>
<td>Top k=5</td>
<td>79.50</td>
<td>25.68/32.32</td>
<td>71.83</td>
</tr>
<tr>
<td>DistilBERT</td>
<td>Top k=40</td>
<td>83.70</td>
<td>24.70/31.21</td>
<td>70.40</td>
</tr>
<tr>
<td></td>
<td>Top k=80</td>
<td>83.70</td>
<td>24.14/30.89</td>
<td>69.83</td>
</tr>
<tr>
<td>LSTM</td>
<td>Nucleus p=0.9</td>
<td>83.40</td>
<td>23.93/30.62</td>
<td>70.55</td>
</tr>
</tbody>
</table>

We show that how the trade-off between diversity and correctness is reflected in the framework of ZEROGEN.

Overall Results Sampling strategies (e.g., top-k sampling and nucleus sampling) are known to be able to generate text with a higher degree of diversity than other decoding strategies (e.g., greedy search) (Fan et al., 2018; Holtzman et al., 2020). Empirical results in Table 2 demonstrate that a more diverse decoding strategy does not always ensure better performance on downstream tasks. For example, the results of the nucleus sampling strategy, which is considered to generate the most diverse data, achieves a performance nearly 6% and 3% lower than the best decoding strategy on both the SQuAD and QNLI datasets, respectively, while greedy decoding strategy could obtain better results than some sampling strategy (e.g., top-k=40, top-k=80 and nucleus sampling).

In contrast, all sampling strategies are superior to the greedy decoding strategy on the IMDb dataset. Regarding the inconsistent better downstream performance of more diverse decoding strategies, we hypothesize that diversity may come at a price, such as generating samples not pertain to the class described in the prompt. Therefore, we access the quality of a dataset from two perspectives: Diversity and Correctness for quantitative analysis of different datasets.

Diversity We follow previous work (Holtzman et al., 2020) and compute Self-BLEU (Zhu et al., 2018) as a metric of diversity. Self-BLEU is calculated by computing the BLEU score of each generated text using all other generations in the evaluation set as references. A lower Self-BLEU score implies higher diversity. We report 4-gram based Self-BLEU in the first part of Table 3. We find that decoding strategies such as top-k and nucleus sampling lead to more diverse generations. This finding is consistent with previous works (Li et al., 2016; Vijayakumar et al., 2016; Welleck et al., 2020; Holtzman et al., 2020).

Correctness Different from the vanilla generation scenario that ends with the generated text, we use the generated text as training dataset for another small model. Therefore, ZEROGEN requires a more emphasis on the correctness of generated text, i.e., whether the generated text pertain to the corresponding class described in the prompt. To access the correctness of a synthetic dataset, we first train a RoBERTa-Large model with standard dataset of each task.

Table 3: Diversity and Correctness evaluation of generated datasets under different decoding strategies for each task. Oracle refers to the standard dataset with human annotations. Diversity is measured by Self-BLEU4, while Correctness is measured by a well-trained RoBERTa-Large model with standard dataset of each task.

<table>
<thead>
<tr>
<th>TAM</th>
<th>Strategy</th>
<th>IMDb</th>
<th>SQuAD</th>
<th>QNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>Greedy</td>
<td>0.30</td>
<td>0.14</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>Top k=5</td>
<td>0.92</td>
<td>0.55</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>Top k=40</td>
<td>0.59</td>
<td>0.29</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td>Top k=80</td>
<td>0.25</td>
<td>0.17</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>Nucleus p=0.9</td>
<td>0.20</td>
<td>0.17</td>
<td>0.12</td>
</tr>
</tbody>
</table>

5Specifically, we randomly sample 1000 generations, each of which is compared with all 999 other generations as references.
Table 4: Results for different prompts on IMDb and SST-2 dev sets. We use GPT2-XL as PLM and DistilBERT as TAM. \(<X>\) and \(<C>\) represents the input sentence and generated movie name respectively. \(P'\) represents minor revised version of \(P\) for text generation. For ZEROGEN, results are reported using 100k training samples. Scores in underline are trained on 10k generated samples, since the prompt \((P'_1 \& P'_2)\) is too weak and cannot generate 100k distinct samples.

```
<table>
<thead>
<tr>
<th>Setting</th>
<th>Id Prompt Type</th>
<th>Label word &lt;Y&gt;</th>
<th>Prompt Type</th>
<th>IMDb</th>
<th>SST-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROMPTING</td>
<td>P_1 &quot;&lt;X&gt;&quot;</td>
<td>great/terrible</td>
<td>-</td>
<td>51.32</td>
<td>51.83</td>
</tr>
<tr>
<td></td>
<td>P_2 &lt;Y&gt; Movie Review: &quot;&lt;X&gt;&quot;</td>
<td>Positive/Negative</td>
<td>Control code</td>
<td>60.36</td>
<td>52.75</td>
</tr>
<tr>
<td></td>
<td>P_3 Task: Write a review for a &lt;Y&gt; movie. Review: &quot;&lt;X&gt;&quot;</td>
<td>good/bad</td>
<td>Control code with task description</td>
<td>54.20</td>
<td>53.50</td>
</tr>
<tr>
<td></td>
<td>P_4 The movie review in &lt;Y&gt; sentiment is &quot;&lt;X&gt;&quot;</td>
<td>positive/negative</td>
<td>Natural language style</td>
<td>80.64</td>
<td>89.22</td>
</tr>
<tr>
<td></td>
<td>P_5 The &lt;Y&gt; movie review is &quot;&lt;X&gt;&quot;</td>
<td>positive/negative</td>
<td>Natural language style</td>
<td>67.60</td>
<td>72.36</td>
</tr>
<tr>
<td>ZEROGEN</td>
<td>P'_1 It was a &lt;Y&gt; movie: &quot;</td>
<td>great/terrible</td>
<td>-</td>
<td>79.36</td>
<td>75.00</td>
</tr>
<tr>
<td></td>
<td>P'_2 &lt;Y&gt; Movie Review: &quot;</td>
<td>Positive/Negative</td>
<td>Control code</td>
<td>60.88</td>
<td>67.43</td>
</tr>
<tr>
<td></td>
<td>P'_3 Task: Write a review for a &lt;Y&gt; movie. Review: &quot;</td>
<td>good/bad</td>
<td>Control code with task specification</td>
<td>83.40</td>
<td>78.90</td>
</tr>
<tr>
<td></td>
<td>P'_4 The movie review in &lt;Y&gt; sentiment is: &quot;</td>
<td>81.84</td>
<td>86.24</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P'_5 The movie review in &lt;Y&gt; sentiment for movie &quot;&lt;C&gt;&quot; is: &quot;</td>
<td>positive/negative</td>
<td>Natural language style</td>
<td>83.40</td>
<td>86.35</td>
</tr>
<tr>
<td></td>
<td>P'_6 The &lt;Y&gt; movie review is &quot;</td>
<td>77.44</td>
<td>77.06</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

4.5 Prompt Engineering in ZEROGEN

The design of prompts can have huge impact on PROMPTING, as pointed by many previous works (Mishra et al., 2021a; Wei et al., 2022). In this section, we investigate how prompt design instruct text generation and affects ZEROGEN’s performance. We examine three commonly used prompt types: (1) Control code (Keskar et al., 2019), (2) Control code with task description, (3) Natural language style. For SST-2 and IMDb, example prompts and corresponding results can be found in Table 4 (check Appendix A for other tasks).

From Table 4, we first observe that natural language prompts are favored by both ZEROGEN and PROMPTING, rather than prompts contain control code. We hypothesize the reason being that during the pre-training process, the majority of text data fed to the PLMs are natural language sentences, and therefore the PLMs do not contain enough knowledge in control code. Moreover, we observe that ZEROGEN is more robust towards different prompts than PROMPTING: for PROMPTING, a minor change from \(P_3\) to \(P'_3\) will lead to a huge drop in accuracy (16.2% drop in IMDb); for ZEROGEN, applying the same prompt revision, the decrement decreases to 9.4%. Compared with PROMPTING which use prompt to directly instruct label words, ZEROGEN use synthesized data as medium to connect PLM and TAM, thus mitigating the sharp change brought by prompts.

To further explore the potential of prompting, we investigate the two-stage conditional prompt inspired by (Schick and Schütze, 2021). In the running example, based on the task characteristic(to generate a movie review), we first generate movie name using prompt [Movie: ""] and then prompt sentence using \(P'_4\). We can find that with the control of movie name, the generated training corpus is more diverse than using \(P'_4\). With the
desirable correctness (see Table 3), the higher diversity leads to a higher accuracy (from 81.84 to 83.40 in IMDb).

The most suitable prompting type in Question Answering and Natural Language Inference tasks has some differences with Text Classification due to different task characteristics. For details, please refer to the Appendix B.

## 5 Related Work

### 5.1 Prompt-based Zero-shot Learning

With manual crafted natural language prompt, large-scale PLMs have shown impressive zero-shot abilities in a wide array of NLP tasks (Radford et al., 2019; Brown et al., 2020). However, current prompt-based zero-shot learning can be unstable: the choice of prompt contributes a lot to the final performance. This motivates researchers to investigate better ways to automatically search and/or manually construct a proper prompt (Jiang et al., 2020a; Shin et al., 2020; Reynolds and McDonell, 2021; Mishra et al., 2021b). To improve the zero-shot generalization across different prompts, another line of work uses a multitask training mixture made up of a large set of different tasks specified in natural language prompts. This induces a model to better generalize to unseen tasks, as well as being more robust to the wording choices of the prompts. (Khashabi et al., 2020; Zhong et al., 2021; Mishra et al., 2021c; Wei et al., 2021; Sanh et al., 2021; Xu et al., 2022). In comparison, we advocate and analyse a new paradigm for prompt-based zero-shot learning via dataset generation, which is complementary to current prompt searching and multi-task pre-training methods.

### 5.2 Dataset Generation with PLMs

Our work also relates to research in generating data with PLMs, which aims to generate a pseudo dataset to enhance model performance. Early efforts achieve this goal with fine-tuned generative models (Anaby-Tavor et al., 2020; Puri et al., 2020; Kumar et al., 2020; Lee et al., 2021). They first fine-tune the generative models using human annotations, the generated data samples are then combined with human annotations to train the models in a semi-supervised fashion. Supervised data generation methods are also studied for building auxiliary tasks (Vu et al., 2021) and dataset creation based on human and machine collaboration (Liu et al., 2022). To reduce the human efforts on data annotation, another line of works explore data generation methods without the need for human annotations. He et al. (2021) uses unsupervised-trained unconditional generative models to synthesize unlabeled data for semi-supervised learning. Without any model training, Wang et al. (2021) propose to directly use unlabeled in-domain examples as prompts to synthesize high-quality training data. Schick and Schütze (2021) explore dataset generation method from scratch for semantic textual similarity task. One concurrent work (Meng et al., 2022) studies dataset generation for text classification and natural language inference tasks. In comparison, we take the dataset generation framework to the extreme, i.e., consider extremely tiny edge models (e.g., LSTM), explore boarder NLP tasks including question answering, and conduct extensive analysis such as decoding strategies and quality evaluation.

## 6 Conclusion and Future Directions

In this paper, we study an extreme instance of dataset generation via PLMs for zero-shot learning. Without any human annotations, we show that an small LSTM can surpass the zero-shot performance of its PLM counterparts (e.g., GPT2-XL), and even outperform the same model trained with human annotations. Despite the demonstrated effectiveness, we discuss several issues we observed when developing ZEROGEN and reveal a substantial room of improvement in future research.

Despite positive results on TC tasks, we find the stability regarding prompt choice of ZEROGEN is still far from satisfactory on NLI tasks. Future work could include multi-task prompt-based pre-training methods (Sanh et al., 2021; Wei et al., 2021).

Furthermore, we observe noisy examples in synthetic dataset on difficult tasks such as NLI and QA, this situation progressively deteriorates when incorporating more diverse decoding strategy (e.g., Nucleus Sampling). Better decoding strategies are needed to ensure the label correctness while preserving the dataset diversity (Massarelli et al., 2020). Besides, methods that learn from noisy labels can be integrated into the training of the tiny task model (Song et al., 2020).

We hope this paper can provide contributions for further exploiting dataset-generation-based zero-shot learning with large pre-trained language models.
References


Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating training data with language models: Towards zero-shot language understanding. CoRR, abs/2202.04538.


Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt


A Experimental Setup

Evaluation Strategy Due to restricted test set access for some datasets (i.e., SQuAD1.1 and SST-2), we held out a small subset (i.e., 10%) of the training set for validation for model trained in supervised setting, and report results on the validation set. For models trained with synthetic dataset in ZEROGEN framework, we also use a portion (i.e., 10%) as the validation set, without accessing to original validation set. For PROMPTING, we directly evaluate on the original validation set.

Implementation Details For dataset generation, we use Nucleus Sampling (Holtzman et al., 2020) with $p = 0.9$ by default as it is considered to be able to generate both fluent and diverse texts (Holtzman et al., 2020). The scale of synthetic dataset is 200k in the main results, and 100k in other analysis experiments. Regarding prompt selection, we manually design a series of prompts for each task, and report results on the best prompt for PROMPTING and ZEROGEN framework. For NLI tasks, we adopt self-debiasing mechanism with a decay constant of 200 (Schick et al., 2021) to ensure that each generated text pair is not only a good fit for a given label, but also not a good fit for other labels (Schick and Schütze, 2021). We removing overly short/long sentences, and sentences without an ending quotation mark.

We implement a LSTM-based model and a DistilBERT model as TAM. For LSTM-based model, we use Adam optimizer (Kingma and Ba, 2015), a learning rate of 1e-4, an embedding dim of 100, and a hidden size of 300. For single sentence classification (TC), we use 1-layer BiLSTM to encode the sentence and use a linear classifier. For sentence-pair classification (NLI), we use 2-layer BiLSTM to encode the sentences to $v_1, v_2$ and pass the concatenated $[v_1; v_2; |v_1 - v_2|; v_1 * v_2]$ to a classifier. For QA tasks, we use the 1-layer BiDAF model. To ensure that TAMs are truly trained from scratch using the synthetic corpus, we random initialize TAMs’ embedding without using any pre-trained word embeddings (e.g., GloVe (Pennington et al., 2014). For DistilBERT, we fine-tune on each dataset with Adam optimizer, with a learning rate of 2e-5, a weight decay of 0.01, and other default hyper-parameters as suggested by HuggingFace Transformers library (Wolf et al., 2019). For all the experiments, we run on a single NVIDIA A100 80G GPU, and generating 200k examples cost 12h on average.

B Additional Results on Prompt Design

For Question Answering tasks, the natural language style prompt is also the most suitable for both PROMPTING and ZEROGEN settings, achieving the highest scores. However, for Natural Language Inference tasks, the most suitable prompts for QNLI and RTE are different. For RTE, the natural language style prompt is best, while the control code prompts perform significantly better than natural language style prompts in QNLI.

C Additional Related Work on Efficient Inference of PLMs

There is a line of works dedicated to improving the inference efficiency of PLMs, including pruning (Wang et al., 2020; Gordon et al., 2020), low-rank factorization (Ma et al., 2019; Noach and Goldberg, 2020; Lan et al., 2020), quantization (Zafrir et al., 2019; Shen et al., 2020; Kim et al., 2021), and knowledge distillation (Jiao et al., 2020; Sanh et al., 2019; Sun et al., 2020). We refer the readers to Xu et al. (2021) for a detailed survey. Concerning privacy, copyright or confidentiality, data-free knowledge distillation (DFKD) (Liu et al., 2021) has attracted appealing attention in computer vision field as it deals with distilling valuable knowledge from well-trained models without requiring to access to the training data. However, similar approaches for NLP are difficult to work due to discrete character of words. Rashid et al. (2021) relax the data-free condition and use out-of-distribution labeled data to train a generator. By contrast, our method generates data with the PLMs (i.e., the teacher), without requiring any pre-defined labeled data. In the literature of knowledge distillation, ZEROGEN framework could produce a student model that achieves superior zero-shot performance the teacher model.

D Generated Examples

We present some qualitative examples for different tasks in Appendix Table 6. Text classification task (SST-2) is relatively simple and concise, the generated samples generally fit the prompts and sentiment polarity well by using descriptive tokens.
about the given movie name and positive/negative sentiment. Take the first case in SST-2 as an example, the generated tokens “action-adventure” and “attractive” are the natural continuations for movie name “The Spiderwick Chronicles (Movie)” and “positive” sentiment in prompt. Although natural language inference tasks are complex, the generated questions (QNLI) and inferences (RTE) could respond to different types of prompts and relate to the given contexts (e.g., the generated question drifts topic for prompt “Information:…Question (answer not in above information)” in QNLI). While the context of the question answering task (SQuAD) is long and contains a lot of information, ZEROGEN could successfully generate question “Who is the one and only true God ?” which is used to response to the pre-set answer “Jehovah”. Overall, these generation examples show that ZEROGEN can generate useful and arbitrary number of training samples that could be used to train TAMs.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Id</th>
<th>Prompt</th>
<th>Label words &lt;Y&gt;</th>
<th>Prompt Type</th>
<th>SQuAD</th>
<th>AdversarialQA</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROMPTING</td>
<td>$P_1$</td>
<td>Context: &quot;&lt;C&gt;&quot;&lt;question&gt;=&quot;&lt;X&gt;&quot;&lt;answer&gt;=&quot;&quot;</td>
<td>Control code</td>
<td>2.69/10.90</td>
<td>1.33/5.61</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_2$</td>
<td>Context: &quot;&lt;C&gt;&quot;&lt;question&gt;=&quot;&lt;X&gt;&quot;&lt;answer&gt;=&quot;&quot;</td>
<td>Control code with task description</td>
<td>3.81/12.1</td>
<td>1.60/5.89</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_3$</td>
<td>The answer to the question &quot;&lt;X&gt;&quot; = &quot;</td>
<td>Control code</td>
<td>4.41/12.60</td>
<td>2.00/5.78</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_4$</td>
<td>The context is: &quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;</td>
<td>Control code</td>
<td>6.33/15.49</td>
<td>2.27/6.71</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_5$</td>
<td>The context is: &quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;</td>
<td>Control code</td>
<td>4.61/13.32</td>
<td>2.13/6.30</td>
<td></td>
</tr>
<tr>
<td>ZEROGEN</td>
<td>$P_1$</td>
<td>Context: &quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;&lt;Y&gt;&quot;&lt;question&gt;=&quot;&quot;</td>
<td>Control code</td>
<td>9.96/9.62</td>
<td>2.87/5.33</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_2$</td>
<td>Context: &quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;&lt;Y&gt;&quot;&lt;question&gt;=&quot;&quot;</td>
<td>Control code</td>
<td>5.84/8.70</td>
<td>2.77/5.21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_3$</td>
<td>&quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;&lt;Y&gt;&quot; is the answer to the question: &quot;</td>
<td>Control code</td>
<td>24.55/29.36</td>
<td>5.30/8.82</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_4$</td>
<td>&quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;&lt;Y&gt;&quot; is the answer to the question: &quot;</td>
<td>Control code</td>
<td>23.58/29.84</td>
<td>5.87/9.58</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$P_5$</td>
<td>&quot;&lt;C&gt;&quot;&lt;answer&gt;=&quot;&quot;&lt;Y&gt;&quot; is the answer to the following question: &quot;</td>
<td>Control code</td>
<td>23.93/30.62</td>
<td>5.97/10.02</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Results for different prompts on Question Answering and Natural language Inference tasks. Results are reported on dev set. For Question Answering, "<C>" represents the input context, "<X>" represents the input question, "<Y>" represents the generated answer, "\n" represents the newline symbol. $P'$ represents minor revised version of $P$ for text generation.
<table>
<thead>
<tr>
<th>Task</th>
<th>Prompt</th>
<th>Sampled Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SST-2</td>
<td>The movie review in positive sentiment for movie “The Spiderwick Chronicles (Movie)” is: &quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>It’s an action-adventure movie about a crime fighter who must face an even bigger foe, then the end of the world, all in a funny, attractive, and moving way.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The movie review in negative sentiment for movie “Fist Fight” is: &quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The action scenes are laughable, the characters lack personality, and the plot is boring.</td>
<td></td>
</tr>
<tr>
<td>QNLI</td>
<td>Information: &quot;In September 1695, Captain Henry Every, an English pirate on board the Fancy, reached the Straits of Bab-el-Mandeb, where he teamed up with five other pirate captains to make an attack on the Indian fleet making the annual voyage to Mocha.&quot; Information: &quot;Gaddafi remained the government’s public face, with the identities of the other RCC members only being publicly revealed on 10 January 1970.&quot; Question (answer in above information): &quot; Why did each of these captains enter the Straits of Bab-el-Mandeb? Question (answer not in above information): &quot; Was the financing received from the international revolution from Germany (4)?</td>
<td></td>
</tr>
<tr>
<td>RTE</td>
<td>“Today about 75% of people live in cities or towns. Art and culture are an important part of France.” Based on the above description, the following sentence is definitely correct: &quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Today about 75% of people live in cities and towns. The most recently documented case was a Sierra Leonean man who died in 2015.</td>
<td></td>
</tr>
<tr>
<td>SQuAD</td>
<td>The context is: “On the other hand, in his study on nine of the Bibles most widely in use in the English-speaking world; Bible scholar Jason BeDuhn, Professor of Religious Studies at the Northern Arizona University, wrote: 201cThe NW [New World Translation] emerges as the most accurate of the translations compared.201d Although the general public and many Bible scholars assume that the differences in the New World Translation are the result of religious bias on the part of its translators, BeDuhn stated: 201cMost of the differences are due to the greater accuracy of the NW as a literal, conservative translation of the original expressions of the New Testament writers.201d He added however that the insertion of the name Jehovah in the New Testament violate[s] accuracy in favor of denominationally preferred expressions for God.” Question (answer not in above information): &quot;Who is the one and only true God? &quot;</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Generated examples for each task. We omit the example for IMDb and Adversarial QA tasks since we use the exactly same prompt as SST-2 and SQuAD task, respectively. The input conditions in each prompt are bolded.